Order constraints for single machine scheduling with non-linear cost*

Christoph Dürr† Oscar C. Vásquez‡

April 24, 2013

Abstract

Typically in a scheduling problem we are given jobs of different lengths p_j and different priority weights w_j, and need to schedule them on a single machine in order to minimize a specific cost function. In this paper we consider the non-linear objective function $\sum w_j C_j^\beta$, where C_j is the completion time of job j and $\beta > 0$ is some arbitrary real constant. Except for $\beta = 1$ the complexity status of this problem is open. Past research mainly focused on the quadratic case ($\beta = 2$) and proposed different techniques to speed up exact algorithms. This paper proposes new pruning rules and generalizations of existing rules to non-integral β. An experimental study evaluates the impact of the proposed rules on the exact algorithm A*.

1 Introduction

In a scheduling problem the goal is to produce a schedule for the given jobs, that minimizes some objective function, usually depending on the job completion times, under problem specific constraints. One natural goal is to minimize the maximum completion time among all jobs, aka *makespan*. Another natural goal is to minimize the average completion time. A common way to combine these two objectives is to minimize the L_β-norm of the completion times, for some constant $\beta > 1$. As an example, the TEX-typesetting system uses the ℓ_3 metric to compute optimal line breaks.

In this paper we consider the more general objective function $\sum_j w_j C_j^\beta$ where w_j is the given priority weight of job j, C_j its completion time and β some fixed positive constant. For convenience we denote the penalty function $f : t \mapsto t^\beta$. We consider the most simple setting, where we are given n jobs, each job j has a processing time p_j and a priority weight w_j, and a schedule consists of an ordering of these jobs on a single machine. Here the completion time of job j is simply p_j plus the sum of the processing times p_i over all jobs that are scheduled before j. There are several motivations to this scheduling problem. Some machines have a learning effect, and their efficiency is increasing during the execution, while some other machines have a wear and tear effect and their efficiency is decreasing during the execution. This can be expressed with values of β respectively smaller or greater than one. Moreover in [13, 7] a particular dynamic voltage scheduling problem optimizing quality of service and energy consumption is reduced to this same objectif function for some $0 < \beta < 1$.

Embarrassingly very little is known about the computational complexity of this problem, except for the special case $\beta = 1$, where scheduling jobs in order of decreasing Smith ratio w_j/p_j leads to the optimal schedule, as has been found out 60 years ago [10]. Most research focused on the quadratic objective function, i.e. for $\beta = 2$, and exact algorithms have been proposed [8, 3, 9, 2, 12].

For branch-and-bound algorithms pruning rules were proposed which reduce the number of nodes in the search graph, having a direct effect on the running time. For example, if we knew,

*Supported by ANR-Netoc

†CNRS, LIP6, Université Pierre et Marie Curie, Paris, France

‡LIP6 and Industrial Engineering Department, University of Santiago of Chile

that without loss of generality in an optimal schedule, job i is never scheduled after job j, then we could eliminate roughly half of the potential orderings, and reduce the number of explored nodes. For example it is known that if $p_j > p_i$ and $w_j < w_i$ then an optimal schedule schedule would never schedule j before job i, which we denote by $i \prec_g j$.

So an extensive literature was devoted to finding stronger rules, which are weaker conditions on the job characteristics that would still imply $i \prec_g j$. The contribution of our paper is to provide new rules and generalize existing rules to arbitrary values of $\beta > 0$. We conclude this paper with an experimental study of the impact of our proposed pruning rules to the performance of an exhaustive search procedure.

2 Pruning rules

We distinguish two kind of properties.

- The *local order property* for jobs i and j, denoted by $i \prec_{\ell(t)} j$, is defined as follows. For any problem instance I including jobs i, j, if in a schedule S for I job j starts at time t and is followed immediately by job i then S is not optimal.
- The *global order property* between jobs i and j, denoted by $i \prec_{g[a,b]} j$, is defined as follows. For any problem instance I including jobs i, j, if in a schedule S for I we have $a \leq C_j - p_j \leq C_i - p_i - p_j \leq b$, then S is sub-optimal, no matter if i, j are adjacent or not[1].

We use the notation $i \prec_g j$ as a shortcut for $i \prec_{g[0,\infty)} j$. In addition $i \prec_{\ell[a,b]} j$ means $i \prec_{\ell(t)} j$ for all $t \in [a, b]$, and $i \prec_\ell j$ stands for $i \prec_{\ell[0,\infty)} j$.

The main result of our paper is that $i \prec_\ell j$ implies $i \prec_g j$. Clearly the proof needs more ingredients than a simple exchange argument since swapping two non-adjacent jobs can increase the cost of the jobs in between.

It was known for $\beta = 2$ that for any job pair i, j there is at most a single time t such that none of $i \prec_{\ell(t)} j$ and $j \prec_{\ell(t)} i$ holds. For completeness we provide a proof for a more general statement in Lemma 1. For this particular timepoint t, exchanging i with j preserves the objective value in any schedule where j starts at time t and is immediately followed by job i. We call this case *degenerate*, and by breaking ties arbitrarily or by disturbing the processing times, we can assume for simplicity in this paper that degenerate cases do not happen.

3 Related work

Previous research focused mainly on the quadratic penalty function, i.e. for $\beta = 2$. Branch-and-bound approaches with pruning rules implying order properties have been proposed, see [12, 2, 9, 1, 3, 11]. In 2000, Mondal and Sen [8] conjectured that $\beta = 2, w_i \geq w_j \wedge w_i/p_i > w_j/p_j$ implies the global order property $i \prec_g j$, and provided experimental evidence that this constraint would significantly improve the runtime of a branch-and-prune search. Recently, Höhn and Jacobs [5] succeeded to prove this conjecture. In addition they improved local and global order conditions and generalized them to integer constants $\beta \geq 2$. An extensive experimental study analyzed the effect of these rules to the performance of the branch-and-prune search. The main contribution of this paper is a proof of Mondal and Sen's conjecture in a larger setting.

We distinguish the following known rules.

Sen-Dileepan-Ruparel [9] for any $\beta > 0$, if $w_i > w_j$ and $p_i \leq p_j$, then $i \prec_g j$.

Höhn-Jacobs-1 [5] for $\beta \in \mathbb{N}$, $\beta \geq 3$, if $w_i/p_i \geq \beta w_j/p_j$ then $i \prec_\ell j$.

Höhn-Jacobs-2 [5] for $\beta \in \mathbb{N}$, $\beta \geq 3$, if $w_i \geq w_j$ and $w_i/p_i > w_j/p_j$ then $i \prec_\ell j$.

Mondal-Sen-Höhn-Jacobs Conjectured in [8], proved in [5]. For $\beta = 2$ the previous rule is enforced by the stronger implication $i \prec_g j$.

[1] From the proof of Theorem 1 it will become clear why we require this lower bound on C_j.

4 Preliminaries

We define the following functions on $t \geq 0$

$$\phi_{ij}(t) = \frac{f(t+p_i+p_j) - f(t+p_j)}{f(t+p_i+p_j) - f(t+p_i)}$$

and

$$\Delta_{ij}(t) = w_i f(t+p_i) + w_j f(t+p_i+p_j) - w_j f(t+p_j) - w_i f(t+p_i+p_j).$$

Note that $\phi_{ij}(t)$ is well defined since by assumption $f : t \mapsto t^\beta$ is strictly increasing and the durations p_i, p_j are non-zero. With these notations in mind we get the following equivalences

$$i \prec_{\ell(t)} j \;\equiv\; \phi_{ij}(t) < \frac{w_i}{w_j} \;\equiv \Delta_{ij}(t) < 0.$$

Note that by definition of ϕ_{ij}, the relation $\prec_{\ell(t)}$ is transitive, as one would expect. It is this function ϕ_{ij} that permits us to analyze algebraically the local order property.

The next lemma shows a connection between the properties of function f and properties of function ϕ_{ij}. We note that the particular form of f is needed, and just being monotone and convex is not sufficient for the claim, as shows the counter-example $f : t \mapsto \max\{t, t^2\}$.

Lemma 1 *Consider the penalty function $f : t \mapsto t^\beta$. If $0 < \beta$, $\beta \neq 1$ and $p_i \neq p_j$ then ϕ_{ij} is strictly monotone, in particular:*

- *If $p_i > p_j$ and $\beta > 1$, then ϕ_{ij} is strictly increasing.*
- *If $p_i < p_j$ and $\beta > 1$, then ϕ_{ij} is strictly decreasing.*
- *If $p_i > p_j$ and $\beta < 1$, then ϕ_{ij} is strictly decreasing.*
- *If $p_i < p_j$ and $\beta < 1$, then ϕ_{ij} is strictly increasing.*

Proof: See Figure 1 for an illustration of the claimed properties. Since $\phi_{ij}(t) = 1/\phi_{ji}(t)$, it suffices to consider the case $p_i > p_j$. For convenience we denote $t_1 = t + p_i + p_j$.

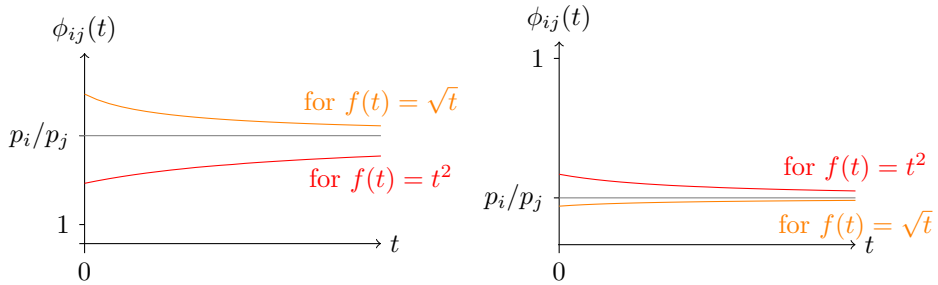

Figure 1: Examples of the function $\phi_{ij}(t)$ for $\beta = 0.5$ and $\beta = 2$, as well as for the cases $p_i > p_j$ and $p_i < p_j$.

Fix an arbitrary $t \geq 0$. We will show that

$$\phi'_{ij}(t) = \frac{[f'(t_1) - f'(t+p_j)][f(t_1) - f(t+p_i)] - [f'(t_1) - f'(t+p_i)][f(t_1) - f(t+p_j)]}{[f(t_1) - f(t+p_i)]^2} > 0.$$

Since the denominator of this fraction is positive, we can focus on the numerator:

$$\begin{aligned}
&[f'(t_1) - f'(t+p_j)][f(t_1) - f(t+p_i)] - [f'(t_1) - f'(t+p_i)][f(t_1) - f(t+p_j)] \\
=&[f'(t_1)f(t_1) - f'(t+p_j)f(t_1) - f'(t_1)f(t+p_i) + f'(t+p_j)f(t+p_i)] \\
&- [f'(t_1)f(t_1) - f'(t+p_i)f(t_1) - f'(t_1)f(t+p_j) + f'(t+p_i)f(t+p_i)] \\
=&- f'(t+p_j)f(t_1) - f'(t_1)f(t+p_i) + f'(t+p_j)f(t+p_i) \\
&+ f'(t+p_i)f(t_1) + f'(t_1)f(t+p_j) - f'(t+p_i)f(t+p_j) \\
=&f'(t_1)f(t+p_j) - f'(t+p_j)f(t_1) + f'(t+p_i)f(t_1) - f'(t_1)f(t+p_i) \\
&+ f'(t+p_j)f(t+p_i) - f'(t+p_i)f(t+p_j).
\end{aligned}$$

Up to factor β this is equivalent to

$$\begin{aligned}
&(t+p_j)^{\beta-1}t^{\beta-1}((t+p_j) - t) + (t+p_i)^{\beta-1}t^{\beta-1}(t - (t+p_i)) \\
&+ (t+p_j)^{\beta-1}(t+p_i)^{\beta-1}((t+p_i) - (t+p_j)) > 0 \\
\equiv&(t+p_j)^{\beta-1}t^{\beta-1}p_j + (t+p_j)^{\beta-1}(t+p_i)^{\beta-1}(p_i - p_j) > (t+p_i)^{\beta-1}t^{\beta-1}p_i \\
\equiv&(p_j/p_i)(1/(t+p_i))^{\beta-1} + (1 - p_j/p_i)(1/t)^{\beta-1} > (1/(t+p_j))^{\beta-1}
\end{aligned}$$

Using a function $h : x \mapsto (1/x)^{\beta-1}$ we reformulate this inequality as

$$(p_j/p_i)h(t+p_i) + (1 - p_j/p_i)h(t) > h(t+p_j).$$

Note that h is a strictly convex function for any $x > 0$ and $\beta > 1$, which implies

$$\lambda h(x_1) + (1-\lambda)h(x_2) > h(\lambda x_1 + (1-\lambda)x_2)$$

for any $\lambda, 0 < \lambda < 1$

We choose $\lambda = p_j/p_i$, $x_1 = t + p_i, x_2 = t$ and obtain

$$\begin{aligned}
\lambda x_1 + (1-\lambda)x_2 &= (p_j/p_i)(t+p_i) + (1 - p_j/p_i)(t) \\
&= tp_j/p_i + p_j + t - tp_j/p_i \\
&= t + p_j.
\end{aligned}$$

In summary we obtain the inequalities

$$(p_j/p_i)h(t+p_i) + (1 - p_j/p_i)h(t) > h((p_j/p_i)(t+p_i) + (1 - p_j/p_i)t) = h(t+p_j).$$

This completes the proof. □

Lemma 2 *Consider the penalty function $f : t \mapsto t^\beta$. For any jobs i, j, we have $\lim_{t\to\infty} \phi_{ij}(t) = p_i/p_j$.*

Proof: We prove this claim using the generalized binomial theorem. Here $\binom{\beta}{k}$ is defined for any real valued $0 < \beta$ as $\beta\,(\beta-1)\cdots(\beta-k+1)/k!$. By definition its value is zero iff β is integral and $0 \le \beta \le k-1$. Then we have

$$(t + p_i + p_j)^\beta = (t+p_j)^\beta + \beta(t+p_j)^{\beta-1}p_i + \sum_{k=2}^{\infty}\binom{\beta}{k}(t+p_j)^{\beta-k}(p_i)^k. \tag{1}$$

Therefore the limit can be expressed as

$$\begin{aligned}
\lim_{t\to\infty}\phi_{ij}(t) &= \lim_{t\to\infty}\frac{(t+p_i+p_j)^\beta-(t+p_j)^\beta}{(t+p_i+p_j)^\beta-(t+p_i)^\beta}\\
&= \lim_{t\to\infty}\frac{(t+p_j)^\beta+\beta(t+p_j)^{\beta-1}p_i+\sum_{k=2}^{\infty}\binom{\beta}{k}(t+p_j)^{\beta-k}(p_i)^k-(t+p_j)^\beta}{(t+p_i)^\beta+\beta(t+p_i)^{\beta-1}p_j+\sum_{k=2}^{\infty}\binom{\beta}{k}(t+p_i)^{\beta-k}(p_j)^k-(t+p_i)^\beta}\\
&= \lim_{t\to\infty}\frac{\beta(t+p_j)^{\beta-1}p_i+\sum_{k=2}^{\infty}\binom{\beta}{k}(t+p_j)^{\beta-k}(p_i)^k}{\beta(t+p_i)^{\beta-1}p_j+\sum_{k=2}^{\infty}\binom{\beta}{k}(t+p_i)^{\beta-k}(p_j)^k}\\
&= \lim_{t\to\infty}\frac{\beta(1+p_j/t)^{\beta-1}p_i+\sum_{k=2}^{\infty}\binom{\beta}{k}\frac{1}{t^{k-1}}(1+p_j/t)^{\beta-k}(p_i)^k}{\beta(1+p_i/t)^{\beta-1}p_j+\sum_{k=2}^{\infty}\binom{\beta}{k}\frac{1}{t^{k-1}}(1+p_i/t)^{\beta-k}(p_j)^k}\\
&= \frac{p_i}{p_j}.
\end{aligned}$$

□

5 Main Results

In [9] it has been shown that if job j has both longer processing time than j and smaller weight than i, then $i \prec_g j$, whereas in [5, 8] some conditions on job pairs that imply the global order property for the quadratic penalty function $f : t \mapsto t^2$ were given. We enforce theses statements using the function ϕ_{ij} and for penalty functions which are not necessarily of the form $f : t \mapsto t^\beta$.

Theorem 1 *Let f be an arbitrary strictly increasing penalty function. Fix two jobs i, j, an interval $[a, b]$ and suppose $i \prec_{\ell[a,b]} j$. If $p_i \le p_j$ or $w_i < w_j$ then $i \prec_{g[a,b]} j$.*

Proof: Suppose that $p_i \le p_j$ or $w_i < w_j$. Let I be an instance containing jobs i, j and S a schedule on I of the form

$$S = AjBiD,$$

for some job sequences A, B, D and $a \le C_j - p_j \le C_i - p_i - p_j \le b$ where C_i, C_j are the respective completion times in S. We have to prove that S is suboptimal. To this end we consider three other schedules $AijBD, AjiBD$ and $AiBjD$.

Denote by $F(S)$ the cost of schedule S, where S is a sequence of jobs. First we show that

$$\max_{t\in[a,b]}\phi_{ij}(t)\left[F(AijBD)-F(AiBjD)\right] > F(AjiBD)-F(AjBiD). \tag{2}$$

For every job $k \in B$ we denote by t_k the completion time of k in the schedule $ABijD$. In (2) we distinguish the contributions of jobs i, j and all jobs $k \in B$. In particular for every job in B we have

$$\max_{t\in[a,b]}\phi_{ij}(t)w_k\left[f(t_k+p_i+p_j)-f(t_k+p_i)\right] \ge w_k\left[f(t_k+p_i+p_j)-f(t_k+p_j)\right] \tag{3}$$

since

$$\phi_{ij}(t_k)w_k\left[f(t_k+p_i+p_j)-f(t_k+p_i)\right] = w_k\left[f(t_k+p_i+p_j)-f(t_k+p_j)\right].$$

We denote by p_B the total processing time over all jobs in B. Then since $f(C_i - p_B) < f(C_i)$ and since $\max_{t\in[a,b]}\phi_{ij}(t) < w_i/w_j$ we have

$$\max_{t\in[a,b]}\phi_{ij}(t)w_j\left[f(C_i-p_B)-f(C_i)\right] > \frac{w_i}{w_j}\cdot w_j\left[f(C_i-p_B)-f(C_i)\right]. \tag{4}$$

Adding (3) and (4) establishes the required inequality (2).

Now for a proof by contradiction, assume that S is optimal. This means that the right hand side of (2) is non-negative and therefore the left hand side is positive. Fix an arbitrary $t \in [a, b]$.

Note that $\phi_{ij}(t) \leq 1$ if $p_i \leq p_j$ by definition of ϕ_{ij} and strict monotonicity of f. Also we have $\phi_{ij}(t) < 1$ if $w_i < w_j$ by case assumption $i \prec_{\ell[a,b]} j$.

This implies by (2)

$$F(AijBD) - F(AiBjD) > F(AjiBD) - F(AjBiD),$$

or equivalently

$$F(AjBiD) - F(AiBjD) > F(AjiBD) - F(AijBD). \tag{5}$$

Optimality of S implies that the left hand side is non-negative and therefore the right hand side of (5) is negative. This contradicts the assumed locality property. □

Theorem 2 *Let f be an arbitrary strictly increasing penalty function, which is either convex or concave. Then for all jobs i, j, $i \prec_\ell j$ and $w_i/p_i > w_j/p_j$ implies $i \prec_g j$.*

Proof: Let i, j be two jobs with $i \prec_\ell j$ and $w_i/p_i > w_j/p_j$. By the previous theorem, we can safely focus on the case $p_i > p_j$.

We will show that for every job sequences A, B, D the schedule $AjBiD$ is suboptimal. The proof is by induction on the number of jobs in B. The basis case $B = \emptyset$ follows simply by the assumption $i \prec_\ell j$.

For the induction step we assume that for *some* job sequences B, D we have that for *every* job sequence A, the schedule $AjBiD$ is suboptimal. The reminder of the proof considers two cases.

Case f is convex: We will show that for every job k and job sequence A the schedule $AjkBiD$ is suboptimal as well. To this end assume $F(AjkBiD) < F(AkjBiD)$ and consider the following equation,

$$\begin{aligned} F(AjkBiD) - F(AikBjD) = \;& F(AjkBiD) - F(AkjBiD) \\ & + F(AkjBiD) - F(AkiBjD) \\ & + F(AkiBjD) - F(AikBjD). \end{aligned}$$

Since by case assumption $F(AkjBiD) > F(AkiBjD)$, to conclude the proof it suffices to show

$$F(AjkBiD) - F(AkjBiD) + F(AkiBjD) - F(AikBjD)$$

is positive. Let t be the total processing time of all jobs in A. Then the later expression can be restated as

$$\begin{aligned} & w_i(f(t + p_k - p_i) - f(t + p_i)) \\ & + w_j\,(f(t + p_j) - f(t + p_k + p_j)) \\ & + w_k(f(t + p_j + p_k) - f(t + p_i + p_k)) \\ = \;& w_i\phi_{ki}(t)(f(t + p_i + p_k) - f(t + p_k)) \\ & - w_j\phi_{kj}(t)(f(t + p_j + p_k) - f(t + p_k)) \\ & + w_k(f(t + p_j + p_k) - f(t + p_i + p_k)) \\ = \;& (w_i\phi_{ki}(t) - w_k)(f(t + p_i + p_k) - f(t + p_k)) \\ & - (w_j\phi_{kj}(t) - w_k)(f(t + p_j + p_k) - f(t + p_k)). \end{aligned}$$

Since f is increasing and $p_i > p_j$ we have

$$f(t + p_i + p_k) - f(t + p_k) > f(t + p_j + p_k) - f(t + p_k).$$

Therefore it suffices to show

$$w_i\phi_{ki}(t) - w_k \geq w_j\phi_{kj}(t) - w_k > 0.$$

The later inequality follows by the assumption $F(AjkBiD) < F(AkjBiD)$. To establish the former inequality we consider the following function

$$g : x \mapsto x\frac{f(t+p_k+x)-f(t+x)}{f(t+p_k+x)-f(t+p_k)},$$

and claim that it is increasing with x. On one hand the fraction

$$\frac{x}{f(t+p_k+x)-f(t+p_k)}$$

represents the slope between the points on the function f evaluated at $t+p_k$ and $t+p_k+x$, and this slope is increasing with x by convexity of f. On the other hand $f(t+p_k+x)-f(t+x)$ is also increasing with x again by convexity. To conclude we observe that

$$w_j\phi_{kj}(t) = \frac{w_j}{p_j}g(p_j) \leq \frac{w_i}{p_i}g(p_j) \leq \frac{w_i}{p_i}g(p_i) = w_i\phi_{ki}(t),$$

where the inequalities used all assumptions on jobs i, j.

Case f is concave: We will show that for every job k and job sequence A the schedule $AjBkiD$ is suboptimal as well. To this end assume $F(AjBkiD) < F(AjBikD)$ and consider the following equation,

$$\begin{aligned} F(AjBkiD) - F(AiBkjD) &= F(AjBkiD) - F(AjBikD) \\ &\quad + F(AjBikD) - F(AiBjkD) \\ &\quad + F(AiBjkD) - F(AiBkjD). \end{aligned}$$

By the assumption above it suffices to show that

$$F(AjBkiD) - F(AjBikD) + F(AiBjkD) - F(AiBkjD)$$

is positive. Let t be the total processing time of all jobs in A and B, and for convenience let be $t_1 = t + p_i + p_j + p_k$. Similar to the previous case we rewrite the expression above as

$$\begin{aligned} & \quad w_i(f(t_1) - f(t+p_j+p_i)) \\ & + w_j(f(t+p_i+p_k) - f(t_1)) \\ & + w_k((f(t+p_j+p_k) - f(t+p_i+p_k)) \\ = & \quad w_i\phi_{ik}(t+p_j)(f(t_1) - t(t+p_j+p_k)) \\ & - w_j\phi_{jk}(t+p_i)(f(t_1) - f(t+p_i+p_k)) \\ & - w_k(f(t_1) - f(t+p_i+p_k)) \\ & + w_k(f(t_1) - f(t+p_j+p_k)) \\ = & \quad (w_i\phi_{ik}(t+p_j) - w_k)(f(t_1) - f(t+p_j+p_k)) \\ & - (w_j\phi_{jk}(t+p_i) - w_k)(f(t_1) - f(t+p_i+p_k)). \end{aligned}$$

By monotonicity of f it suffices to show that

$$w_i\phi_{ik}(t+p_j) > w_j\phi_{jk}(t+p_i).$$

To this end we define again a function

$$g : x \mapsto x\frac{f(t_1) - f(t+p_i+p_j)}{f(t_1) - f(t_1-x)}.$$

This function is increasing with x, because the fraction $x/(f(t_1) - f(t_1-x))$ describes the slope of the line going through the points of f evaluated at t_1 and $t_1 - x$, and by concavity of f, this slope is increasing with x. Therefore we have the following inequalities

$$w_i\phi_{ik}(t+p_j) = \frac{w_i}{p_i}g(p_i) > \frac{w_i}{p_i}g(p_j) > \frac{w_j}{p_j}g(p_j) = w_j\phi_{jk}(t+p_i).$$

This concludes the proof

□

The previous statements permit us to enumerate some conditions on job pairs which imply either a local or a global order property.

Corollary 1 *Consider the penalty function $f : t \mapsto t^\beta$, for $\beta > 0$. Let i, j be two jobs, with $p_i > p_j$.*

If $\beta > 1$	
if $w_i/p_i > w_j/p_j$	*then* $i \prec_g j$
if $\phi_{ij}(0) > w_i/w_j$	*then* $j \prec_g i$
else $\exists t^* : w_i/w_j = \phi_{ij}(t^*)$ *and*	$i \prec_{\ell[0,t^*)} j$
	and $j \prec_{g[t^*,\infty)} i$
if $\beta < 1$	
if $\phi_{ij}(0) < w_i/w_j$	*then* $i \prec_g j$
if $w_i/p_i < w_j/p_j$	*then* $j \prec_g i$
else $\exists t^* : w_i/w_j = \phi_{ij}(t^*)$ *and*	$i \prec_{g[0,t^*)} j$
	and $j \prec_{\ell[t^*,\infty)} i$.

Proof: The proof considers only the case $\beta > 1$, the other case is symmetric. By Lemma 1, ϕ_{ij} is strictly increasing. Therefore $\phi_{ij}(0) > w_i/w_j$ implies $\phi_{ij}(t) > w_i/w_j$ for all $t \geq 0$, that is $j \prec_\ell i$. Now with Theorem 1 we obtain $j \prec_g i$ as required.

For the case $w_i/p_i > w_j/p_j$, by Lemma 1 we can use the following bounds for every $t \geq 0$,

$$\phi_{ij}(t) < p_i/p_j < w_i/w_j,$$

which implies $i \prec_\ell j$. We now apply the Theorem 2 and obtain $i \prec_g j$ as required.

If none of the inequalities holds, by Lemma 1 and continuity of ϕ_{ij} there must be a unique time t^* such that $\phi_{ij}(t^*) = w_i/w_j$. In addition we have $i \prec_{\ell[0,t^*)} j$ and $j \prec_{\ell(t^*,\infty)} i$. Again we can apply Theorem 1 to conclude $j \prec_{g(t^*,\infty)} i$. □

The Figure 2 shows the contribution of our rules for a penalty function $f : t \mapsto t^\beta$.

6 Experimental study

We conclude this paper with an experimental study, evaluating the impact of the proposed rules on the performance of a search procedure. Following the approach described in [5], we consider the Algorithm A* [4]. The search space is the directed acyclic graph consisting of all subsets $S \subseteq \{1, \ldots, n\}$. Note that the potential search space has size 2^n which is already less than the space of the $n!$ different schedules. In this graph for every vertex S there is an arc to $S \backslash \{j\}$ for any $j \in S$. It is labeled with j, and has cost $w_j t^\beta$ for $t = \sum_{i \in S} p_i$. Every directed path from the root $\{1, \ldots, n\}$ to $\{\}$ corresponds to a schedule of an objective value being the total arc cost.

So the goal is to compute the minimum distance between these two vertices. We use the algorithm A* for this, which explores the graph using a priority queue containing arcs pointing to vertices still to be visited. An arc (S', S) has a weight corresponding to the distance from the root to S through this arc plus a basic lower bound of the optimum cost of scheduling S, which we choose to be simply $\sum_{i \in S} w_i p_i^\beta$.

Pruning is done when constructing the list of outgoing arcs at some vertex S. Potentially every job $i \in S$ can generate an arc, but ordering constraints might prevent that. Let j be the label of the arc leading to S (assuming S is not the root). Let $t_1 = \sum_{k \in S} p_k$. Now if $j \prec_{\ell(t_1 - p_i)} i$, then no arc is generated for job $i \in S$. The same thing happens when there is a job $k \in S$ with $i \prec_{g[0,t_1]} k$. In a search tree such a pruning would cut the whole subtree attached to that arc, but in a directed

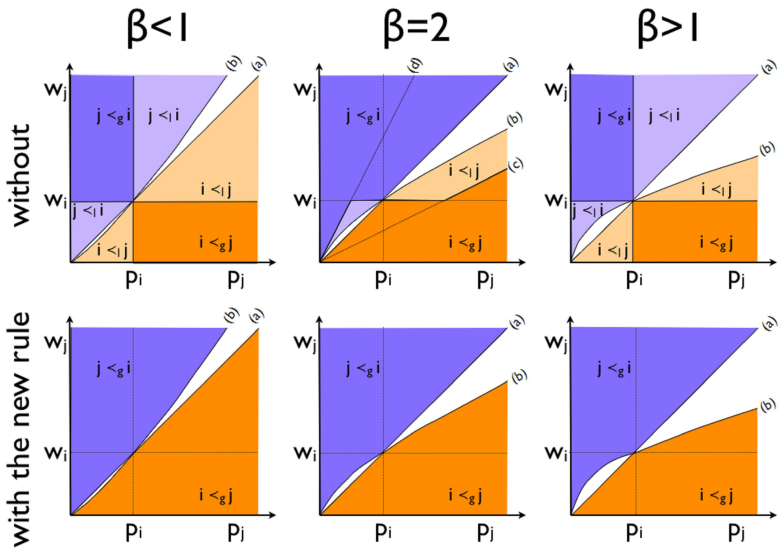

Figure 2: Job j compared to a fixed job i. Labels of particular functions: (a) $w_j = p_j w_i / p_i$, (b) $w_j = w_i((p_i + p_j)^\beta - p_i^\beta)/((p_i + p_j)^\beta - p_j^\beta)$, (c) $w_j = p_j w_i / 2p_i$ and (d) $w_j = 2p_j w_i / p_i$.

acyclic graph the improvement is not so dramatic, as the typical indegree of a vertex is linear in n.

A simple additional pruning is done when remaining jobs to be scheduled form a *trivial* subinstance. By this we mean that all pairs of jobs i, j from this subinstance are comparable with the order $\prec_{\ell[0,t_1]}$. In that case the local order is actually a total order, which describes in a simple manner the optimal schedule for this subinstance. In that case we simply generate a single path from the node S to the target vertex $\{1, \ldots, n\}$.

We consider two variants of the above mentioned algorithm. In the *forward* approach, a partial schedule describes a prefix of some length t of a complete schedule and is extended to its right along an edge of the search tree, and in this variant the basic lower bound is $\sum_{i \in S} w_i(t + p_i)^\beta$. In contrast in the *backward* approach, a partial schedule S describes a suffix of a complete schedule and is extended to its left. Kaindl, Kainz and Radda [6] show experimental evidence that the backward variant generates for some problems less nodes in the search tree, and this fact has also been observed by Höhn and Jacobs [5]. Our own experiments, show that the backward variant shows some advantage over the forward variant, only for large values of α. However when including to the search the rules from this paper the forward variant is better.

Höhn and Jacobs observe that the case when none of the known ordering rules applies to a pair of jobs, is precisely when they have similar but different Smith ratios [5]. Therefore they propose to generate instances with a parameter $\sigma \in [0, 1]$, regulating the expected similarity of the ratios. Their model is as follows. Every job is generated independently. The processing time p_i of a job i is uniformly generated between 1 and 100, and its weight is generated according to $p_i \beta^{N(0,\sigma^2)}$, where N is the normal distribution. So instances become hard to solve not only when the number of jobs is big, but also when σ is close to 0.

We generated instances according to this model, and compared the average number of nodes generated with and without our rules. For $\beta = 2$ we compare our performance with the Mondal-Sen-Höhn-Jacobs rule, while for other values of β we compare with the Sen-Dileepan-Ruparel rule. As a representative of values smaller and respectively greater than 1, we choose $\beta = 0.5$ and $\beta = 3$ for the experiments.

We generated 1000 instances with 20 jobs for different values of σ. A timeout was set, ignoring instances that needed more than a million nodes for the resolution. This never happened when using the new rules, however without them this happened for 70% of the instances for $\sigma = 0.1$, for 20% of the instances for $\sigma = 0.2$ and almost never for larger values of σ.

Over all instances where this timeout was not reached, we averaged the number of nodes explored by the algorithm when not using the new rules and compared it to the average number obtained when using the new rules. This improvement factors are depicted in Figure 3.

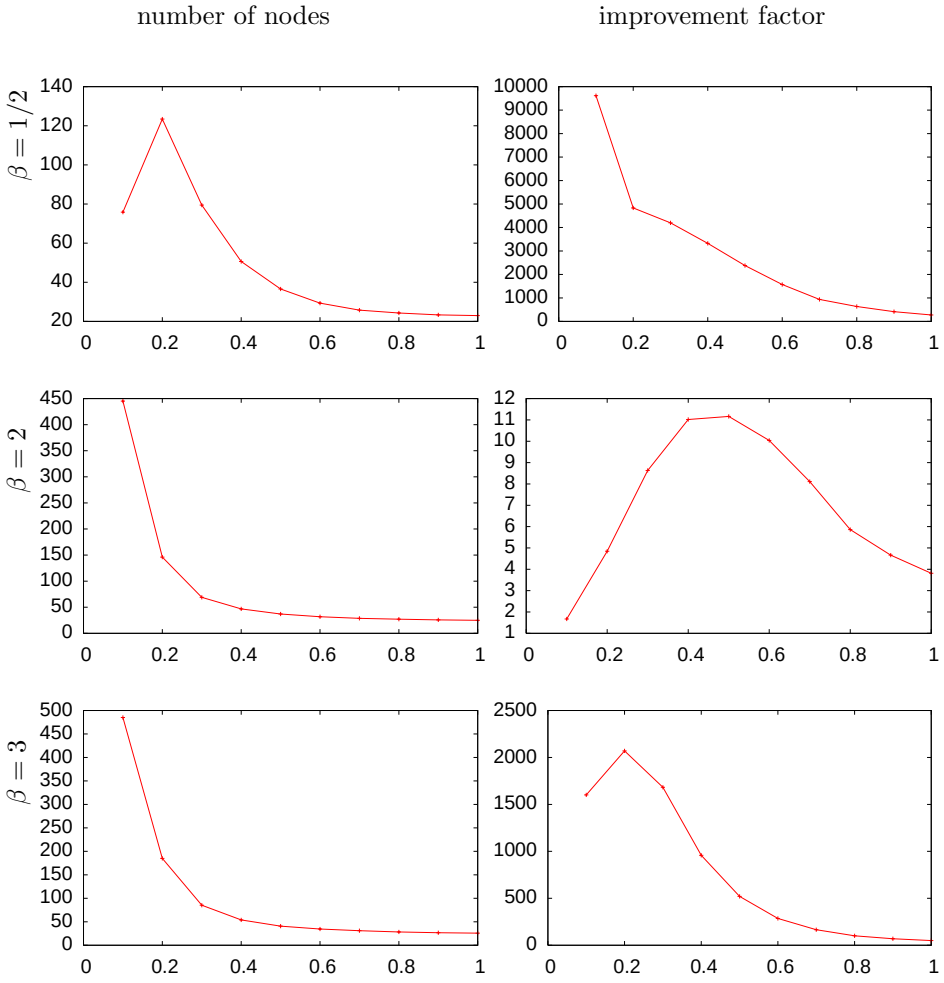

Figure 3: Performance measurements as a function of parameter σ.

In Figure 4 are showed the performance measurements as a function of β. Every point corresponds to the average number of nodes over 100 random instances with $\sigma = 0.3$ and 60 jobs. We can observe that the problem becomes easier when β approaches 1, which is the unique variant of this problem for which a polynomial time algorithm is known.

The implementation was written in C++, single threaded, compiled with gnu C++, and executed on a GNU/Linux PC with an Intel Core2 Duo processor running at at 2.4 GHz and 1Gb RAM. The running time was 8 seconds for the exploration of a million nodes on a 20 job instance.

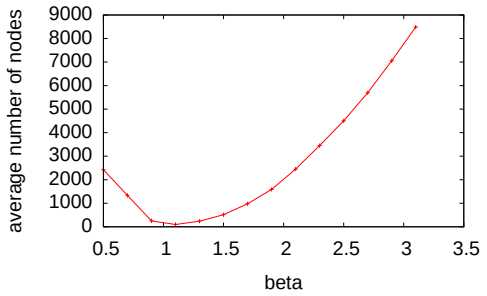

Figure 4: Performance measurements as a function of β.

7 Final remarks

We prove that the relation $i \prec_\ell j$ implies the stronger relation $i \prec_g j$ for all penalty functions of the form $f : t \mapsto t^\beta$. However for interval restricted order properties this implication does not hold. A counter example for $\beta = 2$ consists of 3 jobs $p_i = 13, w_i = 7, p_j = 8, w_j = 5, p_k = 1, w_k = 1$, and where $i \prec_{\ell[0,1]} j$ but the optimal solution is the sequence jki. In this case, $t^* = 19/18$ and the order relation are $i \prec_{\ell[0,t^*)} j$ and $i \prec_{g(t^*,\infty)} j$

We would like to thank anonymous referees for remarks on an earlier version of this paper.

References

[1] B. Alidaee. Numerical methods for single machine scheduling with non-linear cost functions to minimize total cost. *Journal of the Operational Research Society*, pages 125–132, 1993.

[2] P.C. Bagga and K.R. Karlra. A node elimination procedure for Townsend's algorithm for solving the single machine quadratic penalty function scheduling problem. *Management Science*, pages 633–636, 1980.

[3] F.D. Croce, R. Tadei, P. Baracco, and R. Di Tullio. On minimizing the weighted sum of quadratic completion times on a single machine. In *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 816–820. IEEE, 1993.

[4] Peter E Hart, Nils J Nilsson, and Bertram Raphael. Correction to a formal basis for the heuristic determination of minimum cost paths. *ACM SIGART Bulletin*, (37):28–29, 1972.

[5] W. Höhn and T. Jacobs. An experimental and analytical study of order constraints for single machine scheduling with quadratic cost. In *Proc. of the 14th Workshop on Algorithm Engineering and Experiments (ALENEX)*, 2012.

[6] H. Kaindl, G. Kainz, and K. Radda. Asymmetry in search. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 31(5):791–796, 2001.

[7] Nicole Megow and José Verschae. Scheduling on a machine with varying speed: Minimizing cost and energy via dual schedules. Technical Report arXiv:1211.6216, arxiv.org, 2012.

[8] S.A. Mondal and A.K. Sen. An improved precedence rule for single machine sequencing problems with quadratic penalty. *European Journal of Operational Research*, 125(2):425–428, 2000.

[9] T. Sen, P. Dileepan, and B. Ruparel. Minimizing a generalized quadratic penalty function of job completion times: an improved branch-and-bound approach. *Engineering costs and production economics*, 18(3):197–202, 1990.

[10] W.E. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3(1-2):59–66, 1956.

[11] W. Szwarc. Decomposition in single-machine scheduling. *Annals of Operations Research*, 83:271–287, 1998.

[12] W. Townsend. The single machine problem with quadratic penalty function of completion times: a branch-and-bound solution. *Management Science*, pages 530–534, 1978.

[13] Oscar C. Vasquez. Energy in computing systems with speed scaling: optimization and mechanisms design. Technical Report arXiv:1212.6375, arxiv.org, 2012.